Departamento de Engenharia Rural

Centro de Ciências Agrárias

Lógica e Técnica de Programação

Aula 03

- Algoritmos
- Desenvolvendo o raciocínio lógico de programação
- Exemplos de algoritmos
- Exercícios

Algoritmos

- Antes de escrever um programa para resolver um problema, você deve **compreender minuciosamente o problema** e adotar uma **abordagem planejada cuidadosamente** para resolvê-lo.
- Ao escrever um programa, você também deve compreender os **blocos-de-construção** disponíveis e empregar técnicas de construção de programas comprovadas.

Algoritmos

- Qualquer problema computacional pode ser resolvido por meio da execução de uma série de ações (instruções) em uma determinada ordem.

Um procedimento para realizar uma tarefa (resolver um problema) em termos de

- 1) as **ações** a executar
- 2) a **ordem** na qual estas ações são executadas

é chamado **ALGORITMO.**

Algoritmos

O exemplo abaixo demonstra a importância de se especificar corretamente a ordem das ações:

Considere o “algoritmo de acordar e levantar-se” seguido por um executivo para ir trabalhar:

(1) Levante-se da cama; (2) Tire o pijama; (3) Tome banho; (4) Vista-se; (5) Tome café; (6) Vá para o trabalho.

Esta rotina leva o executivo ao trabalho e ele estará bem preparado para tomar decisões críticas.

Algoritmos

Suponha que os mesmos passos sejam executados em uma ordem ligeiramente distinta:

(1) Levante-se da cama; (2) Tire o pijama; (3) Vista-se; (4) Tome banho; (5) Tome café; (6) Vá para o trabalho.

Neste caso, nosso executivo se apresentará encharcado no trabalho. Existem elementos de programação que permitem ao programador controlar a ordem da execução das instruções (ações) de programa. Estas serão apresentadas na próxima aula.

Algoritmos

- Conforme foi visto, os **algoritmos** estão presentes no nosso dia-a-dia, não necessariamente envolvendo aspectos computacionais.

Algoritmos

- Uma **receita de bolo**, uma **partitura musical**, o **manual de utilização de um videocassete** são algoritmos.
  - As ações de uma receita de bolo são do tipo "leve ao forno previamente aquecido", "bata as claras até ficarem em ponto de neve", etc.
  - No caso de uma partitura musical, existem ações que indicam que uma determinada nota musical deve ser executada por determinado tempo
  - O manual de utilização de um videocassete contém ações do tipo "selecione o canal desejado", "aperte a tecla rec", etc.

Algoritmos

- A **execução** de cada um destes algoritmos resulta na realização de alguma **tarefa**:
  - a confecção de um **bolo**;
  - a execução de uma **melodia**;
  - a **gravação** de um programa de televisão.
- Além disso, a execução de cada um deles requer a utilização de equipamentos constituídos de componentes elétricos, mecânicos e eletrônicos, como uma batedeira, um forno, um instrumento musical, uma televisão e um ser humano que seja capaz de interagir com os tais equipamentos para que estes executem as instruções.

Algoritmos

- O conjunto de elementos que interagem para a execução de um algoritmo geralmente é chamado de ***processador***, enquanto que um algoritmo em execução será chamado de ***processo***.
  - No exemplo da partitura musical, o processador é o conjunto {instrumentista, instrumento}
  - No caso de uma receita de cozinha o processador seria o conjunto constituído das panelas, do forno e da cozinheira.
- O resultado do ***processo*** depende dos elementos que compõem o ***processador***.

Algoritmos

- O ***processador*** deve ser capaz de executar as instruções do algoritmo e o ***processo*** deverá **parar** em algum instante para que se tenha a ***realização da tarefa pretendida***.
- Para que estas **duas condições** sejam satisfeitas, é necessário que um algoritmo atenda às seguintes exigências:

1) Ter fim;
2) As instruções devem ser claras, não devendo conter ambiguidades, nem qualquer coisa que impeça sua execução pelo processador (não dar margem à dupla interpretação).
3) Não pode haver dubiedade em relação à próxima ação a ser realizada após a execução de uma determinada instrução.
4) Todas as instruções devem ser executadas num tempo finito.

Algoritmos

- Para exemplificar, considere o seguinte conjunto de instruções, que devem ser executadas sequencialmente:

1) *Sintonize, no videocassete, o canal desejado.*

2) *Insira uma fita no videocassete.*

3) *Acione a tecla rec.*

(Algoritmo 1.0)

Algoritmos

- À primeira vista, o conjunto de instruções anteriormente apresentadas como "***Algoritmo 1.0***" constitui um algoritmo, visto que as exigências estabelecidas para um algoritmo são todas satisfeitas.
- Entretanto, observe que a execução do suposto algoritmo requer a utilização de uma **fita** de videocassete.
- De forma semelhante, uma receita de bolo exige para sua execução os **ingredientes**.
- Isto significa que esses algoritmos necessitam de "**dados**" que deverão ser fornecidos durante suas execuções.
- Estes **dados** constituem as **entradas** dos **algoritmos**.

- Naturalmente, um algoritmo é desenvolvido para que, após a execução de suas instruções, uma determinada **tarefa seja realizada**.
- O resultado da **realização** desta tarefa é conhecida como a **saída** do algoritmo.
  - A **saída** do ***Algoritmo 1.0*** seria a **gravação de um programa** previamente escolhido;
  - A **saída** de uma **receita de bolo** seria o **bolo**;
  - A **saída** de da execução de uma **partitura musical** seria o **som da melodia**.

Algoritmos

- Introduzidos os conceitos de ***entrada*** e ***saída*** de um **algoritmo**, podemos pensar, de forma mais abrangente, que um ***algoritmo*** é um *conjunto de ações* que ao serem executadas em *determinada ordem*, recebendo as ***entradas*** necessárias e compatíveis com as instruções, fornece ***saídas*** adequadas usando *recursos* e *tempo de execução finitos*.

Algoritmos

- É importante observar as saídas produzidas por alguns algoritmos também são sensíveis às entradas.
- Em alguns casos, eventuais problemas podem ser evitados com instruções adicionais e instruções que permitem controlar o fluxo de execução das ações do algoritmo.

Algoritmos

- No ***Algoritmo 1.0***, pode-se perceber um problema: o que aconteceria se a fita que fosse inserida já tivesse sido totalmente utilizada e não houvesse sido previamente rebobinada?
  - A gravação não seria realizada e a saída do algoritmo, para aquela entrada, não ocorreria.

Algoritmos

- Seria necessária então uma instrução que posicionasse a fita no seu início independente da posição em que ela estivesse:

  1) *Sintonize, no videocassete, o canal desejado.*
  2) *Insira uma fita no videocassete.*
  3) *Acione a tecla rr {tecla para rebobinar a fita}.*
  4) *Acione a tecla rec.*

(Algoritmo 1.1)

Algoritmos

- É evidente que a execução da ***instrução 3*** nem sempre resultará em alguma ação efetiva, o que pode ocorrer **se a fita estiver na sua posição inicial**.
- Este problema é resolvido precedendo a execução da instrução por uma **instrução condicional**:

  3) ***Se*** *a fita não estiver rebobinada, acione a tecla rr.*

Algoritmos

- Teria de ser estabelecido algum procedimento que **permitisse ao processador verificar se a fita estaria ou não rebobinada**.
- Isto pode ser feito pelo ser humano que faz parte do processador, ou seja, pela pessoa que está tentando gravar o programa.

Algoritmos

- Assim, o ***Algoritmo 1.1*** seria mais adequado se contivesse as seguintes instruções:

1) *Sintonize no videocassete o canal desejado.*
2) *Apanhe uma fita, verifique se ela está rebobinada e a insira no videocassete.*
3) *Se a fita não estiver rebobinada, acione a tecla rr.*
4) *Acione a tecla rec*

(Algoritmo 1.2)

Algoritmos

- O ***Algoritmo 1.0*** (pelo menos, a maior parte dele) é definido pela própria construção do videocassete e nos é apresentado pelo manual de utilização do aparelho.
- Ou seja, **o algoritmo está pronto e não há necessidade de que se pense**. Não há necessidade de que se **raciocine**.
- Basta que se disponha dos equipamentos necessários e que se seja capaz de executar as instruções.

Aula 03

- Algoritmos
- Desenvolvendo o raciocínio lógico de programação
- Exemplos de algoritmos
- Exercícios

Desenvolvendo o raciocínio lógico de programação

- Estamos interessados em aprender a desenvolver algoritmos que resolvam problemas.
- Nestes casos é necessário pensar. Mais do que isto, é necessário raciocinar. É necessário aprender a raciocinar.
- Como aprender a raciocinar?

Desenvolvendo o raciocínio lógico de programação

- Embora não seja uma definição completa, a ciência do raciocínio é a ***lógica***.
- É esta ciência que estuda os princípios e métodos usados para distinguir os raciocínios **corretos** dos **incorretos**.
- Ao estudar esses métodos, aprende-se a raciocinar (inclusive diante de **situações novas**) e a, portanto, resolver problemas cuja solução não se conhecia.

Desenvolvendo o raciocínio lógico de programação

- De um modo geral, a lógica estuda métodos que permitem estabelecer se um ***argumento de que uma certa afirmativa*** pode ser inferida a partir de outras ***afirmativas*** é ou não correto.
- Por exemplo, a lógica estuda métodos que garantem que a assertiva ***João é racional*** pode ser inferida a partir das afirmações ***todo homem é racional*** e ***João é um homem***.

Desenvolvendo o raciocínio lógico de programação

- Queremos aprender a desenvolver **algoritmos** para resolver **problemas**.
- Queremos aprender a, dado um problema, determinar um **conjunto de instruções** para um **processador** tal que, fornecidas as **entradas**, a **execução** dessas instruções numa determinada **ordem** gere como **saída** a **solução** do problema.

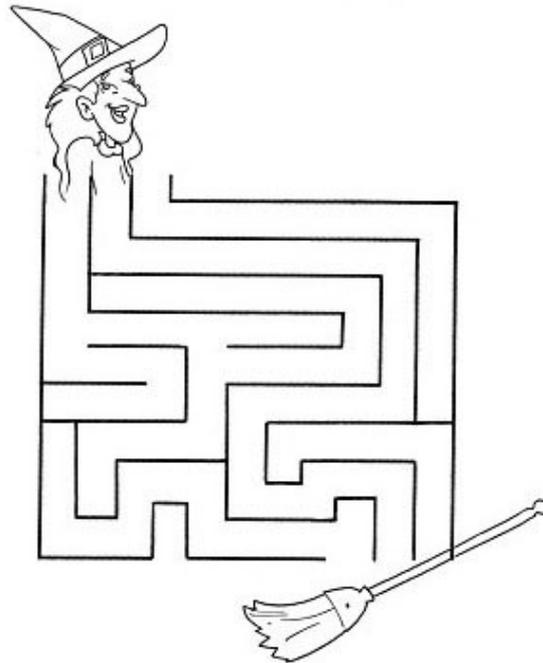
Desenvolvendo o raciocínio lógico de programação

- Embora isto não esteja consagrado ainda pela ciência, nem seja considerado como uma parte da lógica, o ***raciocínio*** que visa ao ***desenvolvimento de algoritmos*** é chamado de ***lógica de programação***.

Desenvolvendo o raciocínio lógico de programação

- A seguir, são apresentados exemplos de **raciocínios lógicos** tipicamente utilizados para resolução de problemas.

Exemplos de raciocínios para resolução de problemas:

- **Exaustão**: quando o número de alternativas possíveis para se atingir um objetivo é pequeno, analisa-se todas elas, uma a uma.

Exemplos de raciocínios para resolução de problemas:

geral → *específico*

- **Dedução:** a dedução consiste em se chegar a uma verdade particular e/ou específica a partir de outra mais geral ou abrangente. Portanto, ao incluirmos um fato específico em outro mais geral, estamos raciocinando por dedução.

❖ Ex. 1:

1) A é sempre igual a B (fato geral, também chamada de *premissa maior*);
2) existe um X que é igual a A (caso particular ou *premissa menor*);
3) logo, este X é igual a B (conclusão).

❖ Ex. 2:

1) Todo número ímpar pode ser escrito como 2n + 1, para qualquer n inteiro;
2) 325 é um número ímpar;
3) logo, 325 pode ser escrito como 2n + 1, se n for igual a 162.

   Ou seja, 2 x 162 + 1 = 325.

Exemplos de raciocínios para resolução de problemas:

número relevante de casos específicos geral

- **Indução**: A indução é o raciocínio que, após considerar um número suficiente de casos particulares, conclui uma verdade geral (*observando casos particulares, isolados, procuramos neles um padrão*).

❖ Ex. 1:

1) Todos os “A”s observados são iguais a B (observação de dados ou fatos isolados);
2) Logo, todo A é igual a B (indução).

Agora, um exemplo numérico:

❖ Ex. 2:

1) Todo número que apresenta o algarismo das unidades igual a 4 é um número par;
2) Logo, 64 é um número par.

Exemplos de raciocínios para resolução de problemas:

- Outra técnica bastante utilizada é se tentar raciocinar a partir de uma solução conhecida de outra questão.
- Para compreender isto considere as duas seguintes questões:
  - imagine uma relação de n números, os quais podem ser referenciados por a_i com $i = 1, 2, ..., n$;
  - queremos somá-los com a restrição de que só sabemos efetuar somas de duas parcelas.

Exemplos de raciocínios para resolução de problemas:

- Para resolver esta questão, podemos pensar em casos particulares:
  - se n = 2, basta somar os dois números;
  - se n = 3, basta somar os dois primeiros e somar esta soma com o terceiro.
  - Naturalmente este raciocínio pode ser reproduzido para n > 3.

Exemplos de raciocínios para resolução de problemas:

- A questão é que a **soma** dos dois primeiros deve estar "**guardada**" para que se possa somá-la com o terceiro, obtendo-se a soma dos três primeiros; esta soma deve ser "**guardada**" para que seja somada com o quarto e assim sucessivamente.
- Para isto podemos estabelecer uma referência à soma "**atual**", a qual será alterada quando a soma com o elemento seguinte for efetuada.
  - Até para somar os dois primeiros, pode-se pensar em somar "**a soma do primeiro**" com o **segundo**.

Exemplos de raciocínios para resolução de problemas:

- Temos então o seguinte algoritmo:
  - 1. Faça $i = 1$.
  - 2. Faça $Soma = a_1$.
  - 3. Repita $n - 1$ vezes as instruções 3.1 e 3.2.
    - 3.1. Substitua i por $i + 1$.
    - 3.2. Substitua Soma por $Soma + a_i$.

Exemplos de raciocínios para resolução de problemas:

- Por exemplo: se n = 5 e a_1 = 8, a_2 = 4, a_3 = 9, a_4 = 13 e a_5 = 7, a execução do algoritmo resultaria nas seguintes ações:
  - 1. i = 1.
  - 2. Soma = 8.
  - 3.1.1. i = 2.
  - 3.2.1. Soma = 8 + 4 = 12
  - 3.1.2. i = 3.
  - 3.2.2. Soma = 12 + 9 = 21.
  - 3.1.3. i = 4.
  - 3.2.3. Soma = 21 + 13 = 34.
  - 3.1.4. i = 5.
  - 3.2.4. Soma = 34 + 7 = 41.

1. Faça i = 1.
2. Faça Soma = a_1.
3. Repita n – 1 vezes as instruções 3.1 e 3.2.
   3.1. Substitua i por i + 1.
   3.2. Substitua Soma por Soma + a_i.

Exemplos de raciocínios para resolução de problemas:

- Como veremos ao longo do curso, este algoritmo é bastante utilizado em programação, sendo mais comum o **primeiro termo** da relação ser "**somado**" dentro da repetição.

Exemplos de raciocínios para resolução de problemas:

- Neste caso, para que o primeiro seja somado, é necessário que a referência **Soma** seja inicializada com 0 (zero), ficando assim o algoritmo:

1. Faça i = 0.
2. Faça Soma = 0.
3. Repita n vezes as instruções 3.1 e 3.2.
   - 3.1. Substitua i por i + 1.
   - 3.2. Substitua Soma por Soma + a_i.

Aula 03

- Algoritmos
- Desenvolvendo o raciocínio lógico de programação
- Exemplos de algoritmos
- Exercícios

Exemplos de Algoritmos

- Resolver problemas matemáticos
  - Algoritmo para determinar a média aritmética de vários números dados;
  - Determinar as raízes de uma equação do segundo grau;
  - Encontrar o máximo divisor comum de dois números dados.

Exemplos de Algoritmos

- Resolver questões genéricas
  - Algoritmo para colocar em ordem alfabética uma relação de nomes de pessoas;
  - Atualizar o saldo de uma conta bancária na qual se fez um depósito;
  - Corrigir provas de um teste de múltipla escolha;
  - Cadastrar um novo usuário de uma locadora, etc.

Aula 03

- Algoritmos
- Desenvolvendo o raciocínio lógico de programação
- Exemplos de algoritmos
- Exercícios

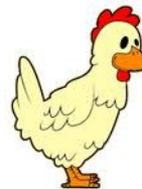
Problemas de lógica

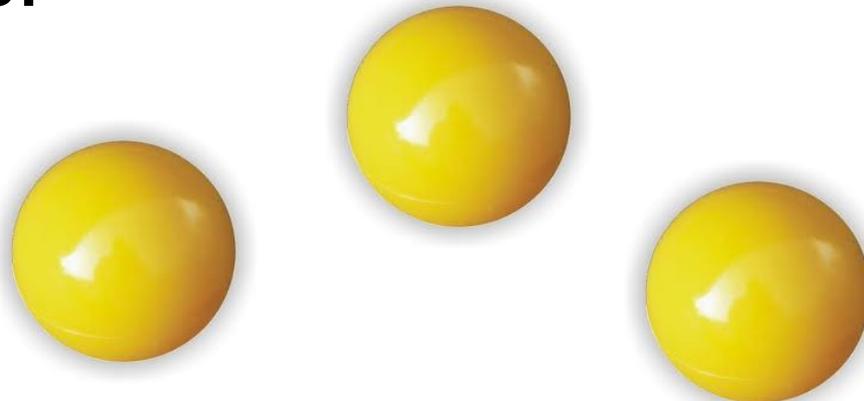
1) (LOGQ01) - Um senhor está numa das margens de um rio com uma raposa, uma dúzia de galinhas e um saco de milho. Ele pretende atravessar o rio com suas cargas, num barco que só comporta o senhor e uma das cargas.

- Escreva uma sequência de ações necessárias para que ele consiga atravessar o rio sabendo que:

  a) A raposa não pode ficar sozinha com as galinhas, pois estas seriam comidas pela raposa.

  b) As galinhas não podem ficar sozinhas com o milho.

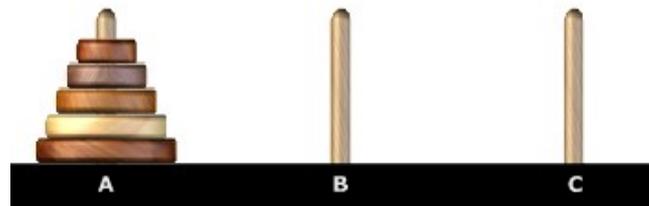
  c) A raposa pode ficar sozinha com o milho.

Problemas de lógica

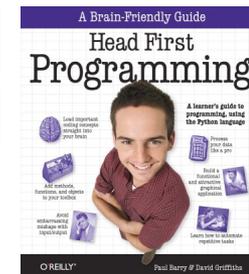
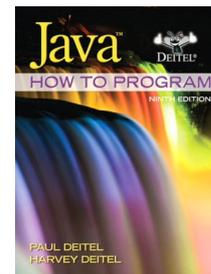
2) (LOGQ02) Dispõe-se de três esferas idênticas na forma, sendo duas delas de mesmo peso e a terceira de peso maior. A questão é descobrir qual a esfera de peso diferente, realizando-se apenas uma pesagem numa balança de dois pratos.

Problemas de lógica

3) (LOGQ03) como obter exatamente 4 litros de água dispondo de dois recipientes com capacidades de 3 litros e 5 litros?

Problemas de lógica

4) (LOGQ04) Dispõe-se de três torres A,B,C dispostas nesta ordem, onde inicialmente a torre A possui um certo número de peças empilhadas de forma que uma peça está sempre acima de outra com diâmetro maior que o seu.
O problema é transportar todas as peças da torre A para a torre C, podendo-se apenas mover uma peça por vez (de uma torre para outra) e com a restrição de que a cada passo uma peça não pode ser colocada sobre outra de diâmetro menor.

- Resolver com 3, 4 e 5 peças.

Referências

- BORLAND. Turbo Pascal, versão 7.0. [S.L.] : Borland International, Inc.,1992.
- DEITEL, P. J.; DEITEL, H.M.; Java: How to program, 9th ed, Ed. Prentice-Hall, 2011. ISBN: 978-0-13-257566-9.
- FARRER, H.; BECKER, C. G.; FARIA, E. C.; MATOS, H. F.; et al. Algoritmos estruturados. 3ed, Ed. LTC, 1999. ISBN: 9788521611806.
- FARRER, H.; BECKER, C. G.; FARIA, E. C.; MATOS, H. F.; et al. Pascal estruturado. 3ed, Ed. LTC, 1999. ISBN: 9788521611745.
- GUIMARÃES, A. M.; LAGES, N. A. C.; Algoritmos e estruturas de dados. 1ed, Ed. LTC, 1994. ISBN: 9788521603788.
- GRIFFITHS,D., BARRY,P., Head First Programming – A learner's guide to programming using the Python language, O´Reilly, 2009, 406p.
- http://wwwusers.rdc.puc-rio.br/rmano/processo.html
- Velloso, F. C.; Informática: Conceitos Básicos. 7ed, Ed. Campus, 2004. ISBN: 9788535215366.

Respostas problemas de lógica

- (LOGQ1 – Travessia do rio) – Resposta:
  1) Atravesse as galinhas.
  2) Retorne sozinho.
  3) Atravesse a raposa.
  4) Retorne com as galinhas.
  5) Atravesse o milho.
  6) Retorne sozinho.
  7) Atravesse as galinhas.

Respostas problemas de lógica

- (LOGQ2 – Esfera de peso diferente) – Resposta:
  1) Escolha duas esferas.
  2) Coloque cada uma das esferas escolhidas num dos pratos da balança.
  3) Se a balança ficar equilibrada, forneça como resposta a esfera não escolhida; caso contrário, forneça como resposta a esfera do prato que está num nível mais baixo.

Respostas problemas de lógica

4 = 3+1

Passo	Estado
1. Encha o recipiente de 3 litros.	1. (A, 3), (B, 0).
2. Transfira o conteúdo do recipiente de 3 litros para o recipiente de 5 litros.	2. (A, 0), (B, 3).
3. Encha o recipiente de 3 litros.	3. (A, 3), (B, 3).
4. Com o conteúdo do recipiente de 3 litros, complete o recipiente de 5 litros.	4. (A, 1), (B, 5).
5. Esvazie o recipiente de 5 litros.	5. (A, 1), (B, 0).
6. Transfira o conteúdo do recipiente de três litros para o recipiente de 5 litros.	6. (A, 0), (B, 1).
7. Encha o recipiente de 3 litros.	7. (A, 3), (B, 1).
8. Transfira o conteúdo do recipiente de 3 litros para o recipiente de 5 litros.	8. (A, 0), (B, 4).

Ou 4 = 5 - 1

Passo	Estado
1. Encha o recipiente de 5 litros.	1. (A, 0), (B, 5).
2. Com o conteúdo do recipiente de 5 litros, encha o de 3 litros.	2. (A, 3), (B, 2).
3. Esvazie o recipiente de 3 litros.	3. (A, 0), (B, 2).
4. Transfira o conteúdo do recipiente de 5 litros para o recipiente de 3 litros.	4. (A, 2), (B, 0).
5. Encha o recipiente de 5 litros.	5. (A, 2), (B, 5).
6. Com o conteúdo do recipiente de 5 litros, complete o recipiente de 3 litros.	6. (A, 3), (B, 4).

Respostas problemas de lógica

- (LOGQ4 – Torre de Hanoi)
  - Assumindo P1, P2, P3, P4 e P5 como as peças, onde os diâmetros de P5>P4>P3>P2>P1
  - Considere também T1, T2 e T3 como as torres 1, 2 e 3 respectivamente;
  - Por fim, denotando o movimento da peça i para a torre j como (Pi,Tj)(i = 1,2,3,4,5 e j = 1, 2, 3) temos as seguintes soluções com o menor número possível de passos:

Respostas problemas de lógica

- (LOGQ4 – Torre de Hanoi) – 3 peças – 7 passos
  1. (P1,T2)
  2. (P2,T3)
  3. (P1,T3)
  4. (P3,T2)
  5. (P1,T1)
  6. (P2,T2)
  7. (P1,T2)

Respostas problemas de lógica

- (LOGQ4 – Torre de Hanoi) – 4 peças – 15 passos
  1. (P1,T2)
  2. (P2,T3)
  3. (P1,T3)
  4. (P3,T2)
  5. (P1,T1)
  6. (P2,T2)
  7. (P1,T2)
  8. (P4,T3)
  9. (P1,T3)
  10. (P2,T1)
  11. (P1,T1)
  12. (P3,T3)
  13. (P1,T2)
  14. (P2,T3)
  15. (P1,T3)

Respostas problemas de lógica

- (LOGQ4 – Torre de Hanoi) – 5 peças – 31 passos

1. (P1,T3)
2. (P2,T2)
3. (P1,T2)
4. (P3,T3)
5. (P1,T1)
6. (P2,T3)
7. (P1,T3)
8. (P4,T2)
9. (P1,T2)
10. (P2,T1)
11. (P1,T1)
12. (P3,T2)
13. (P1,T3)
14. (P2,T2)
15. (P1,T2)
16. (P5,T3)
17. (P1,T1)
18. (P2,T3)
19. (P1,T3)
20. (P3,T1)
21. (P1,T2)
22. (P2,T1)
23. (P1,T1)
24. (P4,T3)
25. (P1,T3)
26. (P2,T2)
27. (P1,T2)
28. (P3,T3)
29. (P1,T1)
30. (P2,T3)
31. (P1,T3)